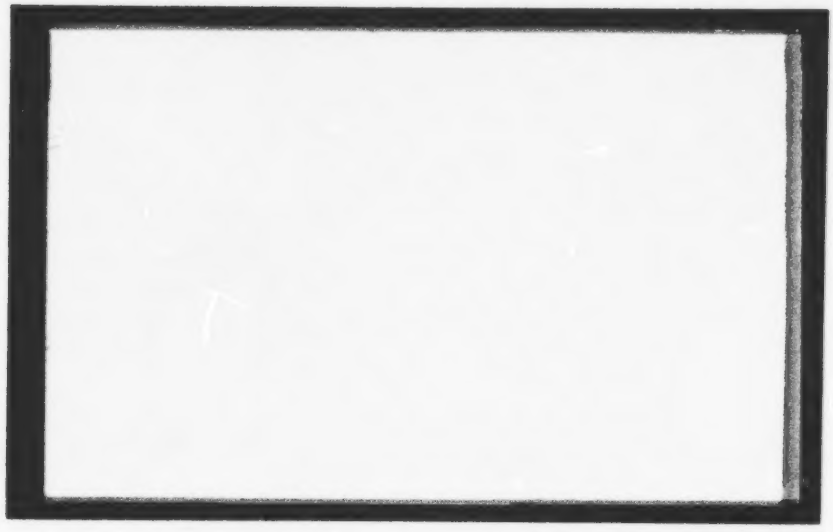
UNIVERSITÉ DE MONTRÉAL



CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

METHODS FOR ROUTING WITH TIME WINDOWS

by

Jacques Desrosiers, François Soumis
Martin Desrochers and Michel Sauvé

Université de Montréal

Centre de recherche sur les transports - Publication #317
Ecole des Hautes Etudes Commerciales -
Cahier du GERAD #G-83-05

Février 1983, révisé en décembre 1983

Dépôt légal - 1er trimestre 1984

ABSTRACT

Consider a set of trips where each trip is specified a priori by a place of origin, a destination, a duration, a cost and a time interval within which the trip must begin. The trips may include visits to one or more specific points. Our problem is to determine the number of vehicles required together with their routes and schedules, so that each trip begins within his given time interval, while the fixed costs related to the number of vehicles, and the travel costs between trips are minimized. The problem is a generalization of the m-travelling salesman problem.

We compare numerical results for 3 algorithms developed by our research team:

1) column generation on a set partitioning problem solved by simplex and branch-and-bound; columns are generated by a shortest path algorithm with time constraints on the nodes.

2) adaptation of the Carpaneto-Toth algorithm for the asymmetric travelling salesman problem: solution of network problems by relaxing scheduling constraints, and branch-and-bound on flow variables.

3) solution of network problems by relaxing scheduling constraints and branch-and-bound based on dividing the time windows.

METHODS FOR ROUTING WITH TIME WINDOWS

1. The Problem

Consider a set of trips, where each trip is specified a priori by a place of origin, a destination, a duration, a cost, and a time interval $[a_i,b_i]$ within which the trip must begin. The trips may include visits to one or more specific points. A "route" is defined as a sequence of trips made by the same vehicle. The problem is to determine the number of vehicles required, together with their routes and schedules, so that each trip begins within his given time interval, while the fixed costs related to the number of vehicles, and the travel costs between trips are minimized.

An arc (i,j) is defined as the movement of a vehicle from the end of trip i to the beginning of trip j. The duration is t_{ij} and its associated travel cost is c_{ij}. Arc (i,j) can be defined only if the following relation holds:

$$a_i + t_{ij} \leq b_j \qquad (1)$$

Let P be the set of trips and I the set of arcs satisfying relation (1). A network [N;A] is constructed in which the set of trips P forms the set of nodes along with a node s for exit from the vehicle depot and a node t representing entrance to the depot. The arcs are formed by the union $I \cup \{s\}\times P \cup P\times\{t\}$. The fixed cost of a vehicle is assigned to the depot exit arcs $\{s\}\times P$.

Two types of variables are used in the mathematical formulation: binary flow variables x_{ij} $(i,j) \in A$ taking the value 1 when arc (i,j) is used by a vehicle, and time variables t_i, $i \in P$ associated with the starting times of trips.

The optimal routes are the solution of the following problem:

$$\text{Min} \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{2}$$

$$\sum_{j\in N} x_{ij} = \sum_{j\in N} x_{ji} = 1 \qquad i \in P \tag{3}$$

$$x_{ij} \geq 0 \qquad (i,j) \in A \tag{4}$$

$$x_{ij} > 0 \Rightarrow t_i + t_{ij} \leq t_j \qquad (i,j) \in I \tag{5}$$

$$a_i \leq t_i \leq b_i \qquad i \in P \tag{6}$$

Relations (2), (3) and (4) form a simple routing problem without scheduling constraints. This is a minimum cost flow problem having an integer solution. Relations (5) describe the compatibility requirements between the routes and the schedules, while constraints (6) establish the time intervals within which trips must begin.

It can be shown that there exists an optimal integer solution to the routing problem with scheduling constraints defined by relations (2) to (6). However, this optimal integer solution cannot be obtained directly by linear programming as relations (5) are not linear constraints. They can however be rewritten in linear form:

$$t_i + t_{ij} - t_j \leq (1 - x_{ij})\, M_{ij} \qquad (i,j) \in I \tag{7}$$

with $M_{ij} \geq b_i + t_{ij} - a_j$. This formulation is equivalent only if x_{ij} is a binary variable. In the special case where $[a_i, b_i] = [0, |P|-1]$ for $i \in P$ and $t_{ij} = 1$ for $(i,j) \in I$, we may set $M_{ij} = |P|$ and constraints (7) become the sub-tour elimination constraints proposed by Miller, Tucker and Zemlin (1960) for the travelling salesman problem:

$$t_i - t_j + |P| x_{ij} \leq |P| - 1 \qquad (i,j) \in I \tag{8}$$

2. The Literature

One possible approximate solution method is to replace each variable t_i by a set of binary variables associated with each decision to begin a trip i or not at a number of discrete points within the time interval. This method

has been used by Levin (1971) for air transportation, and by Swersey and Ballard (1982) for school transportation; they frequently obtained an optimal integer solution using the simplex algorithm without integrality constraints. Note, however, that they restricted the objective function to include only the number of vehicles, after noting that it was more difficult to obtain integrality when travel costs were also included. Other methods have been proposed by Bodin et al (1981) but they require some additional approximations.

Branch-and-bound methods have been most commonly used to solve the same type of problem optimally. A relaxed problem is solved and branching is carried out to eliminate infeasible solutions. Several relaxations have been proposed; they present different compromises between solution costs and quality of approximation. For the asymmetric travelling salesman problem, Carpaneto and Toth (1980) relaxed subtour elimination constraints to obtain an easily solvable assignment problem, and they improved on Bellmore and Malone's (1971) branching on the flow variables to break the subtours. This method has been adapted to our more general problem; when constraints (5) and (6) are very restrictive the gap between the assignment problem and the original problem is considerable and the branch-and-bound tree to be explored may become very large (Desrosiers, Soumis, Desrochers, Sauvé, 1983). To reduce this gap and the size of the tree on related problems, Fisher and Jaikumar (1981) have proposed Lagrangian relaxation while Christofides, Mingozzi and Toth (1981) have proposed state-space relaxation which uses intermediate relaxations close to the original problem starting from shortest path or spanning tree problems.

In this paper, we mainly describe two approaches. In section 3, we solve the linear version of the set partitioning problem (an equivalent formulation of the original problem) by column generation. We take advantage of the network structure in the subproblem and integrality is recovered by branch-and-bound. In section 4, we relax constraints (5) and (6) to obtain a network problem, and we examine a new type of branching based not on the flow variables x_{ij}, but on the division of the time intervals $[a_i, b_i]$.

3. Linear Relaxation of the Set Partitioning Formulation

This method is presented using flow variables on the routes rather than on the trips. A route satisfying the scheduling constraints (5) and (6) is specified by the trips forming it, and their starting times. We introduce the following notation:

Ω : the set of routes r satisfying the scheduling constraints

y_r: a binary variable; 1 if route r is used, 0 otherwise

δ_{ri}: binary constant; 1 if route r includes trip i, 0 otherwise

c_r: cost of route r; the sum of the costs of the arcs making up the route

A set partitioning problem is formulated in which a set of routes satisfying the scheduling constraints and including each trip exactly once is constructed at minimum cost.

$$\text{Min} \sum_{r\in\Omega} c_r y_r \quad \text{subject to} \quad \sum_{r\in\Omega} \delta_{ri}\, y_r = 1 \qquad i \in P \tag{9}$$

$$y_r \text{ binary} \qquad r \in \Omega$$

Because of the large number of routes in Ω, the set partitioning problem is solved by column generation. A new column of minimum marginal cost is generated by solving the following shortest path problem with scheduling constraints defined using the dual variables σ_i of the linear relaxation of the set partitioning problem:

$$\text{Min} \sum_{(i,j)\in A} (c_{ij} - \sigma_i)\, x_{ij} \tag{10}$$

$$\sum_{j\in N} x_{ij} = \sum_{j\in N} x_{ji} \qquad i \in P \tag{11}$$

$$\sum_{j\in P} x_{sj} = \sum_{j\in P} x_{jt} = 1 \tag{12}$$

$$x_{ij} > 0 \Rightarrow t_i + t_{ij} \le t_j \qquad (i,j) \in I \tag{13}$$

$$a_i \le t_i \le b_i \qquad i \in P \tag{14}$$

$$x_{ij} \text{ binary} \qquad (i,j) \in A \tag{15}$$

3.1 Subproblem Solution

The subproblem is solved by dynamic programming. Recall that in the classical shortest path problem, the Ford-Bellman algorithm assigns a single label to each node representing the cost (π_i) which is gradually improved. For the problem with scheduling constraints, the routes must be compared in terms of arrival times at the nodes as well as costs: at each node i a label (arrival time at node i, cost of the route) is defined. Several labels have to be stored at node i as they may be required to calculate the labels of other nodes according to the following optimality principle: "If r is a minimum cost route from among all the routes from s to j arriving at j at time t_j or before, and if (i,j) is its terminal arc, then the sub-route of r from s to i is a minimum cost route from among all routes from s to i arriving at i at time $t_j - t_{ij}$ or before".

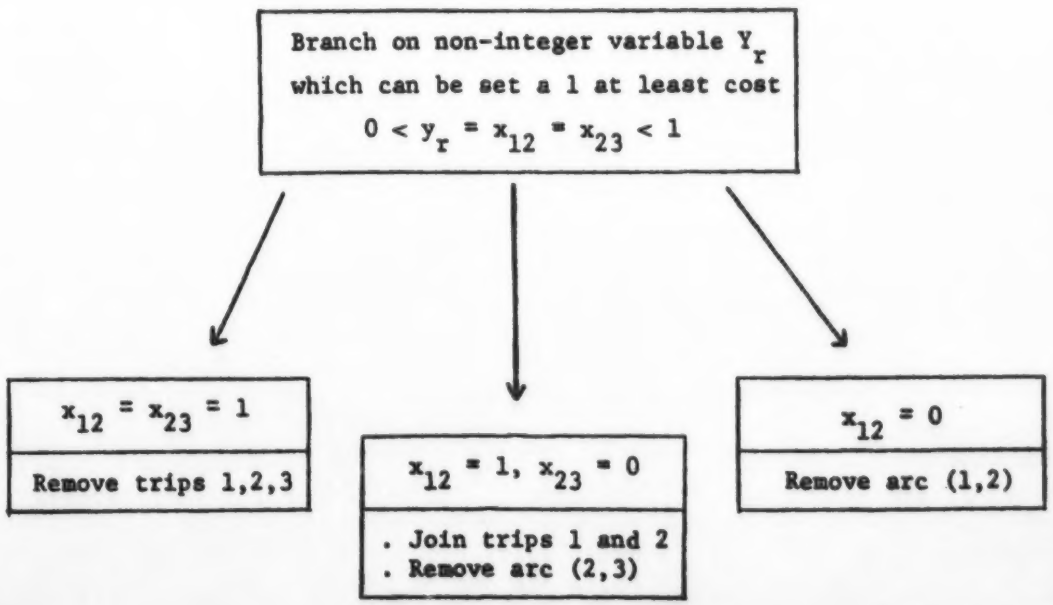
The algorithm stores at each node the label (t_i, π_i) until a less costly route arriving at i at time t_i or before is found. A suitable data structure and an efficient strategy for the management of the lists of labels in this problem are presented in Desrosiers, Pelletier, Soumis (1983). This solution method for the subproblem may produce routes with cycles if the network contains negative cost cycles satisfying the scheduling constraints. If a route with a cycle is constructed, its duration and cost will be bounded because of the time interval constraints. The route is added to the master problem with $\delta_{ri} > 1$ for the trips covered more than once.

3.2 Solution of the Master Problem

The simplex algorithm is used to solve the master problem. This method gives the marginal costs σ_i necessary for column generation and enables easy reoptimization each time new columns are generated. The probability of reaching an integer solution spontaneously with the simplex method is fairly high for this type of problem; we increase this probability by using cuts which require that the number of vehicles and the total travel time be rounded up to the next integer. When the solution is not integer, a branch-and-bound method is used with column generation to solve the problem at each node of the tree.

The branching at a node takes place on the non-integer variable y_r which can be set at 1 at least additional cost, according to the estimates given by the marginal costs. Bellmore and Malone's (1971) method developed for the travelling salesman problem is used for the branching. This is carried out not by fixing the y_r variables, but by setting at 0 or 1 the k variables x_{ij} associated with the arcs which make up the route r. The 2^k ways of fixing these variables are grouped into k + 1 branches including respectively 1, 1, 2, 4,..., 2^{k-1} ways. The first branch fixes the k arcs at 1. The second fixes the first k-1 arcs at 1 and the last at 0. The third fixes the first k-2 arcs at 1 and the next one at 0, and so on until the k+1st branch where the first arc is set at 0. This branching procedure retains in each branch a master problem and subproblem of the same type as those initially defined. Branching by setting $y_r = 0$ would have destroyed the shortest path structure of the subproblems. Figure 1 shows an example in which branching is carried out on a route made up of trips 1, 2 and 3 in that order. At each node of the tree, the covering problem is reoptimized generating new routes as necessary. In the branching tree, the left-most node is always examined first. This ensures that an initial integer solution is rapidly obtained. The cost of this solution is very close to and often equal to the cost of the LP relaxation. Frequently, only a few nodes are required to complete the branching. This solution method is presented in detail in Desrosiers, Soumis, Desrochers (1983).

Figure 1: Branching tree for the LP relaxation


Branch on non-integer variable Y_r which can be set a 1 at least cost
$0 < y_r = x_{12} = x_{23} < 1$
$x_{12} = x_{23} = 1$
Remove trips 1,2,3
$x_{12} = 1$, $x_{23} = 0$
. Join trips 1 and 2
. Remove arc (2,3)
$x_{12} = 0$
Remove arc (1,2)

4. Network Relaxation with Branch-and-Bound on Time Windows

This solution method relaxes the scheduling constraints (5) and (6), and solves a network problem at each node of the tree. The optimal solution of each network problem is integer and involves the construction of a set of routes and sometimes some subtours. The schedule t_i of each trip $i \in P$ must be determined to check whether constraints (5) and (6) are satisfied.

A route of m trips is specified by a sequence of network nodes s, i(1), i(2),...,i(m), t where s and t represent the depot. A subtour of m trips is specified by i(1), i(2),..,i(m), i(m+1) where the network nodes i(1) and i(m+1) are identical. The first trip i(1) of a subtour is chosen so that $a_{i(1)} = \min_{1 \le r \le m} \{a_{i(r)}\}$. As equation (5) allows waiting before the beginning of a trip, the time variables are evaluated as follows:

$$t_{i(1)} = a_{i(1)}$$

$$t_{i(r)} = \max \{a_{i(r)}, t_{i(r-1)} + t_{i(r-1), i(r)}\} \qquad 2 \le r \le m$$

with an additional calculation at node i(1) of a subtour:

$$t_{i(m+1)} = t_{i(m)} + t_{i(m), i(1)} \tag{16}$$

When there are no subtours and the time window constraints are satisfied for all time variables, the solution of the network problem is optimal for the original problem. Otherwise, a branching rule based on a partitioning of the time intervals is established. Two cases are considered:

Case 1: For a route or a subtour of m trips, there is at least one time variable from among $t_{i(1)}, t_{i(2)}, \ldots, t_{i(m)}, t_{i(m+1)}$ which does not satisfy the upper bound of time window constraints.

Case 2: For a subtour of m trips, all the time variables $t_{i(1)}, t_{i(2)}, \ldots$ $t_{i(m)}$, and $t_{i(m+1)}$ satisfy the time window constraints.

4.1 Branching to Satisfy Time Window Constraints

Consider a route or a subtour as in case 1. Let $i(\gamma)$ be the first trip for which the time interval constraint is not satisfied and $\delta = t_{i(\gamma)} - b_{i(\gamma)} > 0$ be the excess time; note that $\gamma \le m$ for a route while $\gamma \le m+1$ for

a subtour. Let $i(\alpha)$ be the last trip before $i(\gamma)$ for which $t_{i(\alpha)} = a_{i(\alpha)}$:

$$\alpha = \max \{ 1 \le r \le \gamma - 2 \mid t_{i(\alpha)} = a_{i(\alpha)} \} \tag{17}$$

The latest times $t^{-1}_{i(r)}$ which would have to have been assigned to trips $i(r)$ preceding trip $i(\gamma)$ so that it could begin at time $t_{i(\gamma)} = b_{i(\gamma)}$ are calculated as follows:

$$t^{-1}_{i(r)} = t_{i(r)} - \delta \qquad \alpha \le r \le \gamma - 1 \tag{18}$$

Obviously, $t^{-1}_{i(\alpha)} = a_{i(\alpha)} - \delta < a_{i(\alpha)}$ does not satisfy the time window constraint; find $i(\beta)$ the last trip before $i(\gamma)$ for which $t^{-1}_{i(\beta)}$ does not satisfy the time interval constraint:

$$\beta = \max \{ \alpha \le r \le \gamma - 2 \mid t^{-1}_{i(r)} < a_{i(r)} \} \tag{19}$$

We therefore have $1 \le \alpha \le \beta \le \gamma - 2$ and the time interval $[a_{i(r)}, b_{i(r)}]$ can be divided into a left interval $L_{i(r)}$ and a right interval $R_{i(r)}$ at the value $t^{-1}_{i(r)}$ for $\beta + 1 \le r \le \gamma - 1$. To simplify the description of the partition associated with this division of the time intervals, and supposing that the t_{ij} are integer, we introduce the following notation:

$$[a_{i(r)}, b_{i(r)}] = [a_{i(r)}, t^{-1}_{i(r)}] \cup [t^{-1}_{i(r)} + 1, b_{i(r)}]$$

$$= L_{i(r)} \cup R_{i(r)} \qquad \beta + 1 \le r \le \gamma - 1$$

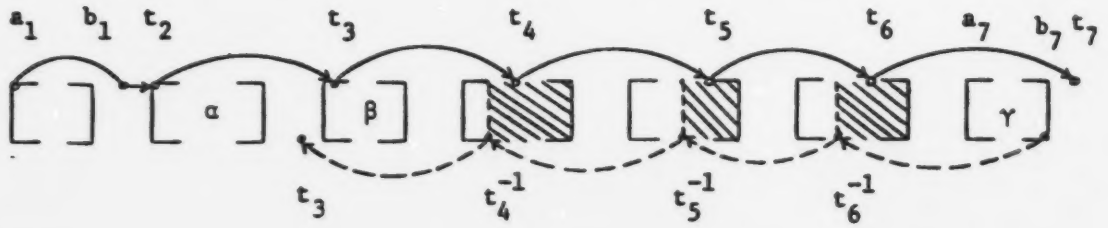
There are $\gamma - \beta - 1$ intervals to be divided into two parts i.e. $2^{\gamma-\beta-1}$ possibilities grouped into $\gamma - \beta$ branches. The partition is described in table (1).

Table 1: Partition of the time intervals

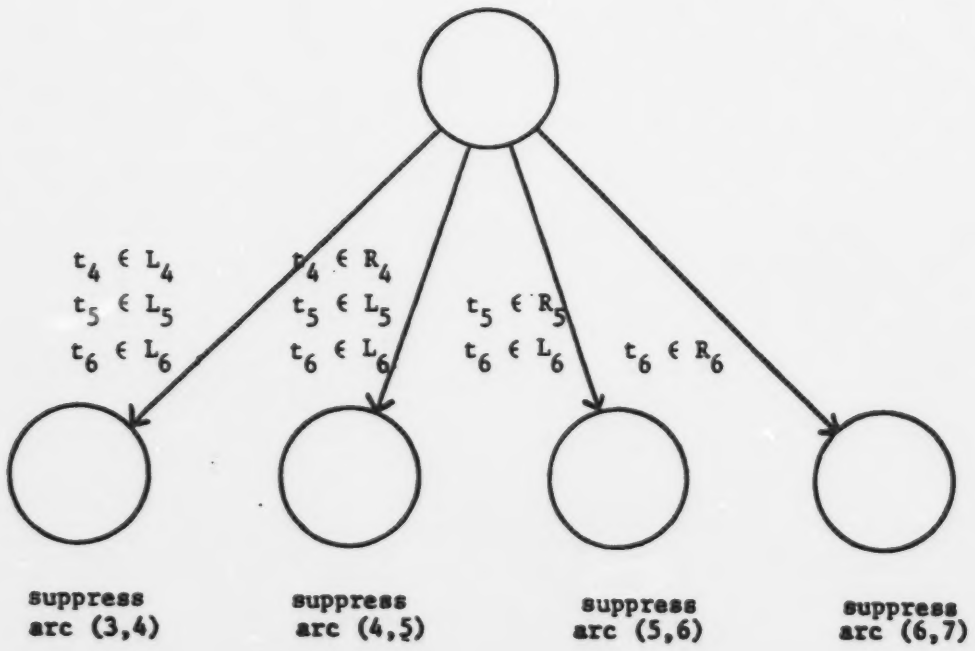
Branch	$t_{i(r)} \in L_{i(r)}$	$t_{i(r)} \in R_{i(r)}$	arc suppressed
1	$r = \beta + 1, \ldots, \gamma - 1$		$(i(\beta), i(\beta+1))$
2	$r = \beta + 2, \ldots, \gamma - 1$	$r = \beta + 1$	$(i(\beta+1), i(\beta+2))$
3	$r = \beta + 3, \ldots, \gamma - 1$	$r = \beta + 2$	$(i(\beta+2), i(\beta+3))$
:			
$\gamma - \beta$		$r = \gamma - 1$	$(i(\gamma-1), i(\gamma))$

This partition is similar to the partition on binary variables used in the column generation branching method; however, no flow variables are set at 1. Each branch suppresses an arc of the route or subtour considered so that this route or subtour is no longer feasible in the new network problem. The reduction of the time intervals also eliminates other network arcs by relation (1). Figure 2 gives an example in which $\alpha = 2$, $\beta = 3$, $\gamma = 7$, and 4 branches are formed.

Figure 2: Branching tree to satisfy time window constraints



$$[a_i, b_i] = [a_i, t_i^{-1}] \cup [t_i^{-1} + 1, b_i] = L_i \cup R_i$$



4.2 Branching to Eliminate Subtours

For a subtour in case 2 containing m trips, we have $t_i(r) \in [a_{i(r)}, b_{i(r)}]$, $1 \leq r \leq m$ and $t_{i(m+1)} \leq b_{i(1)}$. This is eliminated by creating several families of branches so that in each family, this subtour no longer satisfies the time window constraints. For the first family, $t_{i(1)}$ is restricted to the interval $[a_{i(1)}, a_{i(1)} + (t_{i(m+1)} - t_{i(1)}) - 1]$ where $t_{i(m+1)} - t_{i(1)}$ is the duration of the subtour. This subtour no longer satisfies the time interval constraint for the trip $\gamma = m + 1$ and the branches described in the previous section are obtained. For $t_{i(1)} \in [a_{i(1)} + (t_{i(n+1)} - t_{i(1)}), b_{i(1)}]$, the times assigned to the trips in the subtour are recalculated and the analysis is repeated (case 1 or case 2). For the last family of branches (case 1), it is possible that the first arc of the subtour may be illegal i.e. may no longer satisfy relation (1) with the reduced interval for i(1): in this special case, the values $\alpha = \beta = 1$, $\gamma = 2$ are obtained and the number of intervals to be divided is equal to $\gamma - \beta - 1 = 0$. The first arc is suppressed and the family only contains a single branch.

4.3 Branching Strategy

The objective function (2) optimizes the number of vehicles used, and for an equal number of vehicles minimizes the sum of travel times. In our problems, the function has the following form:

$$\text{Min } Z_1 = C * \sum_{j \in P} x_{sj} + \sum_{(i,j) \in I} t_{ij} x_{ij} \qquad (20)$$

where C is the fixed cost per vehicle. In our tests, the network relaxation was never feasible and the gap was often fairly large in terms of travel time and sometimes even in terms of the number of vehicles. To obtain a first feasible solution, it is necessary to descend many levels in the tree, and the number of vehicles used usually goes above the optimal number. Improvement in the objective function is therefore slow, and based mainly on travel times, although large gains can be made in terms of the number of vehicles. We therefore decided to limit this approach to the optimization of the number of vehicles. The secondary travel time term is however retained as it helps reduce the number of illegal routes.

When the network solution is infeasible, the choice of partition is first carried out on the subtours, and then on the illegal routes. However, the subtours satisfying the scheduling constraints are dealt with last as these may generate several families with several branches each; in this way, restrict the number of nodes in the tree.

For an infeasible solution at a node in the branch-and-bound tree, let z be the value of the objective function. For each branch created, an arc (i,j) is eliminated and a lower bound $\bar{z}$ on the value of the optimal solution of the new problem is given by:

$$\bar{z} = z + \min_{j' \neq j} \{\bar{c}_{ij'}\} + \min_{i' \neq i} \{\bar{c}_{i'j}\} \quad . \tag{21}$$

If only the number of vehicles is included in the objective function, then

$$\bar{z} - z = 0 \text{ or } 1 \quad . \tag{22}$$

When the solution at a node of the tree contains several subtours in illegal routes, branching is carried out by choosing the partition which contains the smallest number of branches without increasing the bound, and in case of equality, the partition with the smallest number of branches. Finally, nodes in the tree are explored according to the LIFO strategy, by choosing the node having the best bound amongst those created.

Feasibility was improved by calculating for each arc $(i,j) \in I$, the probability p_{ij} of feasibility, and replacing the secondary term by the logarithm of the product of these probabilities:

$$\text{Min } Z_2 = C * \sum_{j \in P} x_{sj} - \sum_{ij \in I} x_{ij} \ln p_{ij} \tag{23}$$

Assuming that t_i, $i \in P$ is a discrete random variable, the probability that arc (i,j) is feasible is given by:

$$p_{ij} = \begin{cases} 1 & \text{if } b_i + t_{ij} \leq a_j \\ \dfrac{(b_j - (b_i + t_{ij}) + 1)}{(b_j - a_j + 1)} & \text{if } a_j < b_i + t_{ij} \leq b_j \\ \dfrac{1}{(b_i + t_{ij} - b_j + 1)(b_j - a_j + 1)} & \text{if } b_i + t_{ij} > b_j \end{cases} \tag{24}$$

In fact, p_{ij} is the proportion of pairs $(t_i + t_{ij}, t_j)$ satisfying the relation $t_i + t_{ij} \leq t_j$ in the interval $[a_j, b_j]$ if $b_i + t_{ij} \leq b_j$, or in the rectangle $[b_j - t_{ij}, b_i] \times [a_j, b_j]$ if $b_i + t_{ij} > b_j$.
The new objective function contains an approximation of the relaxted constraints (5) and (6) and the solution of the network problem generally contains far fewer illegal routes. This solution method is presented in detail in Desrosiers, Soumis, Desrochers, Sauvé (1983).

5. Testing

The two algorithms were used to solve 8 school transportation problems. Each problem is labelled according to the number of trips, followed by AM or PM to indicate whether morning or afternoon transport is being considered. AM and PM problems involve the same trips but the school schedules are different. School finishing times are spread out over a larger time period than starting times. The number of arcs between trips is therefore greater in the afternoon and the networks are very different (Table 2).

Table 2: Description of problems

PROBLEM SIZE	NUMBER OF SCHOOLS	DURATION OF TRIPS			WIDTH $(b_i - a_i)$			DURATION OF ARCS			NUMBER OF ARCS		TIME PERIOD	
		MIN	MAX	AV.	% OF 0	MAX	AV.	MIN	MAX	AV.	AM	PM	AM	PM
48	6	8	33	18.7	15%	9	4.6	1	29	10.9	579	620	80	110
100	46	13	76	34.4	34%	22	7.1	1	56	19.0	1137	2088	119	166
151	55	8	76	32.9	31%	22	7.8	1	56	18.5	2620	4900	121	166
223	65	8	76	32.6	31%	22	7.8	1	56	18.3	6273	10792	131	166

24 variants of these problems have been constructed by replacing the interval $[a_i, b_i]$ by $[a_i, b_i + \Delta]$, with Δ values of 5, 10, 15 and 20 minutes. Each new problem of this type is identified by the original problem name followed by + Δ (e.g. 48 AM + 20). 6 variants with fixed schedules where all trips i must begin at specific times a_i are identified by adding $t_i = a_i$ to the problem label (e.g. 48 AM $t_i = a_i$).

Table 3: Set Partitioning results with Bellmore Malone's Branching

PROBLEM NAME	NUMBER OF ARCS	LP RELAXATION			CUTS		BEST INTEGER SOLUTION				FINAL RESULTS				
		BUS	COST	COMP. TIME[1]	BUS	COST	BUS	COST	NODE[2]	OPTIMAL	GAP[3]	NODES[4]	COLUMNS[5]	LABELS[6]	COMP. TIME
48AN t_1=a_1	490	20	185	3.1/1.6	--	--	20	185	1	YES	--	1	129	1	3.1/1.6
48AN	579	17	191	5.3/3.1	--	--	17	191	6	YES	--	6	163	1.5	16.5/4.1
48AN+5	669	15	213	13/5.3	--	--	15	213	2	YES	--	2	174	2	19.1/5.7
48AN+10	764	13.3	--	23/15	14	209	14	209	2	YES	--	2	262	5	64/18
48AN+15	876	12.4	--	33/25	13	205	13	207	6	?	2	26	424	9	375/43
48AN+20	999	11.7	--	44/56	12	210	12	210	1	YES	--	1	307	14	84/71
48PN t_1=a_1	512	24	212	2.6/1.3	--	--	24	212	1	YES	--	1	121	1	2.6/1.3
48PN	620	17	320	13/3.2	--	--	17	320	1	YES	--	1	156	2	13/3.2
48PN+5	711	14	380.6	25/7.8	14	381	14	387	15	?	6	50	431	3.4	687/24
48PN+10	800	13	370.2	36/16	13	371	13	373	3	?	2	30	402	5.4	526/40
48PN+15	888	11.9	--	35/29	12	363	12	364	3	YES	1	10	365	7.8	274/60
48PN+20	999	11.1	--	34/51	12	313	12	318	9	?	5	16	404	11	668/87
100AN t_1=a_1	787	70	286	5.5/4.6	--	--	70	286	1	YES	--	1	181	1	5.5/4.6
100AN	1137	63	390	12/14	--	--	63	390	1	YES	--	1	238	1.6	12/14
100AN+5	1522	58.5	--	18/20	59	394	59	394	1	YES	--	1	260	1.8	31/21
100AN+10	1793	54.5	--	24/24	55	476	55	479	11	?	3	153	420	2	756/71
100AN+15	2093	51	515	53/36	--	--	51	515	2	YES	--	2	299	2.5	67/36
100AN+20	2505	46	608	59/57	--	--	46	608	3	YES	--	3	337	3.5	80/63
100PN t_1=a_1	1703	59	464	4.9/5.8	--	--	59	464	1	YES	--	1	175	1	4.9/5.8
100PN	2088	50.5	--	33/22	51	571	51	571	1	YES	--	1	290	2.1	35/24
100PN+5	2478	46	644	41/32	--	--	46	644	1	YES	--	1	308	2.1	41/32
100PN+10	2797	43.5	--	74/44	44	635	44	635	2	YES	--	2	347	2.6	185/47
100PN+15	3018	39.5	--	106/151	40	690	40	691	16	?	1	58	498	3	779/102
100PN+20	3338	37	736	176/76	--	--	37	736	7	YES	--	7	455	3.1	379/102
151AN t_1=a_1	1535	108	406	11/10	--	--	108	406	1	YES	--	1	261	1	11/10
151AN	2620	94	560	45/49	--	--	94	560	1	YES	--	1	343	2	43/49
151AN+5	3572	87.3	--	90/80	88	595	88	595	1	YES	--	1	427	2.3	128/82
151AN+10	4311	82	734	263/119	--	--	82	734	1	YES	--	1	454	3	263/119
151AN+15	5108	73.5	--	183/68	74	781	74	781	3	YES	--	3	518	3.2	319/192
151AN+20	6068			(memory > 200 000$_8$)											
151PN t_1=a_1	3769	92	654	14/18	--	--	92	654	1	YES	--	1	298	1	14/18
151PN	4900	75	850.3	215/67	75	851	75	851	2	YES	--	2	473	2	307/69
151PN+5	5787	69.5	--	354/104	70	903	70	903	6	YES	--	6	525	2.3	648/122
151PN+10	6529	65.5	--	637/133	66	943	66	943	1	YES	--	1	546	3.5	557/135
151PN+15	7077	58	1059	633/261	--	--	58	1059	4	YES	--	4	629	4.2	877/269
151PN+20	7795			(memory > 200 000$_8$)											

(1) COMPUTER TIME ON CDC 173, SECONDS IN SIMPLEX/SECONDS IN SHORTEST PATH.

(2) NUMBER OF NODES EXPLORED IN BRANCH-AND-BOUND BEFORE REACHING THE BEST INTEGER SOLUTION.

(3) GAP IN SECONDS BETWEEN THE COST OF THE BEST INTEGER SOLUTION AND THE LP RELAXATION WITH CUTS.

(4) NUMBER OF NODES EXPLORED IN BRANCH-AND-BOUND.

(5) TOTAL NUMBER OF COLUMNS GENERATED.

(6) MEAN NUMBER OF LABELS PER NODE IN THE SHORTEST PATH WITH TIME WINDOWS.

5.1 Results of the Set Partitioning Formulation

To test the problems, the fixed vehicle cost was set at 10 000 to ensure that the number of vehicles is first minimized and for equal numbers of vehicles, the travel time between trips is minimized. The 223 trip problems cannot be solved within the $200\ 000_8$ word memory limit as the simplex algorithm used stores the inverse matrix explicitly (Land, Powell, 1973). Original problems of up to 151 trips were solved very easily. There is no gap between the cost of the linear relaxation of the set partitioning problem and the cost of the optimal solution and the branch-and-bound tree reduces to a small number of nodes. These problems are not however trivial as the number of vehicles is reduced by 10% to 30% as compared with similar problems in which the start times are fixed. The calculations increase when the time intervals are widened but most problems can still be solved optimally even if the network problem is very different from the fixed schedule problem. When $\Delta = 20$, the problems are sufficiently different that the number of vehicles can be reduced by 30% to 50% as compared with the fixed schedule solution. For solutions where optimality is unproven, the number of buses is minimal, and the gap in travel time between trips is less than 1.6%. This percentage is reduced to 0.5% if total vehicle travel is considered (Table 3).

5.2 Results for the Network Relaxation with Branch-and-Bound on Time Windows

When the number of vehicles and the travel time were optimized simultaneously, the two small problems 48AM and 48PM were solved in 1.6 seconds (11 nodes) and 2.0 seconds (19 nodes) respectively. For the other problems, the method sometimes produced good solutions but never completed the enumeration of the tree. Results for optimization of the number of vehicles only on the original problems (Δ=0) are given here. Two variants are presented (Table 4): these use different secondary terms in the objective function, and the same branching rule.

Variant 1: $$Z_1 = C * \sum_{j \in P} x_{sj} + \sum\sum_{ij \in I} t_{ij} x_{ij}$$

Variant 2: $$Z_2 = C * \sum_{j \in P} x_{sj} - \sum\sum_{ij \in I} x_{ij} \ln p_{ij}$$

Table 4: Network relaxation results with branching on the time windows

PROBLEM NAME	NETWORK RELAXATION BUSES	ILLEGAL ROUTES	FIRST SOLUTION BUSES	NODE	BEST SOLUTION BUSES	NODE	NODES EXPLORED	MAXIMUM LEVEL	COMPUTER TIME (SEC) CDC 171
48AM (1)	17	1	17	3	17	3	3	2	1
(2)		1	17	2	17	2	2	1	1
48PM	17	3	17	21	17	21	21	20	3
		0	17	1	17	1	1	0	1
100AM	63	6	64	19	63	212	212	18	49
		1	63	2	63	2	2	1	3
100PM	48	7	51	18	51	18	188	17	51
		4	51	7	51	7	53	7	17
151AM	93	11	95	37	94	42	865	38	> 390
		5	95	24	94	36	893	32	> 390
151PM	72	12	76	37	75	41	137	38	74
		6	75	13	75	13	120	12	71
223AM	134	14	136	61	135	62	111	61	> 93
		11	136	45	135	47	47	45	44
223PM	104	16	109	65	107	86	467	64	> 390
		10	107	17	107	17	323	16	> 390

(1): $Z_1 = C * \sum_{j \in P} x_{oj} + \sum\sum_{ij \in I} t_{ij} x_{ij}$;

(2): $Z_2 = C * \sum_{j \in P} x_{oj} - \sum\sum_{ij \in I} x_{ij} \ln P_{ij}$;

For the 8 problems the time intervals are sufficiently small that the solution of the network problem includes no subtours. The gap between the optimal number of vehicles and that found by the network relaxation is sometimes fairly large reaching 3 vehicles for problems 100PM, 151PM, and probably also 223PM for which the best solution of 107 vehicles has not been proved optimal. The first variant solved only 4 problems to optimality while the second solved 6. For the two unsolved problems, 151AM and 223PM, the best solutions had not been proved optimal in 390 seconds with the second variant. The solution of 94 vehicles found for 151AM is however optimal; this problem was solved in less than 100 seconds using the column generation algorithm. For all the problems studied, the second variant dominates the

first according to all possible criteria: it produces a greater reduction in the number of illegal routes in the network relaxation, it finds a better or equivalent initial feasible solution and a better final solution while exploring fewer nodes in the tree, the total size of the tree is smaller, and it requires less computer time.

6. Conclusion

The linear relaxation of the set partitioning formulation gives a very good lower bound on the number of vehicles and the travel time. This bound is excellent even for wide time windows and it does not deteriorate as the number of trips increases. Furthermore when there is no integrality gap, the simplex method often gives the integer solution directly or after fixing several variables. On the other hand, when there is an integrality gap, even if this is small, the Bellmore-Malone branching method is very weak and the tree is very large. For example, for the problem 100AM+10, the gap was three minutes and optimality had not been obtained after 155 nodes had been explored.

The network relaxation rapidly calculates a lower bound using very little memory, but the quality of this bound diminishes as the size of the windows and the number of trips increases. This approach used with the Bellmore and Malone branching method solved optimally problems of 128 and 158 trips with very narrow time windows (Soumis, Desrosiers, Desrochers, 1983). For these problems the time windows were however still wide enough to allow a reduction of 5% in the number of vehicles compared with the fleet required to complete the trips according to a fixed schedule. On the other hand, the method could not find the optimal solution to the 48AM problem even after 150 seconds of computer time; the best solution was 25 minutes of travel time away from the optimum. This problem has time windows which allow a reduction of 15% in the number of vehicles as compared with the fixed schedule case. If the Bellmore-Malone branching method is replaced by branching on the time intervals in the network relaxation, it is

possible to solve the 48AM and 48PM problems optimally with respect to fleet size and travel time in less than 2 seconds. These results demonstrate the superiority of the time interval branching method. For larger problems where the bound is less good and gives differences of up to 3 vehicles, branching on time intervals still produces solutions with the minimum number of vehicles.

The set partitioning method is the best for solving problems with large time windows. For these problems, the network relaxation which gives integer solutions, and then uses branching to satisfy the scheduling constraints is not efficient. The gap is too big and the tree to be explored is very large. By dealing with the scheduling constraints during the construction of the routes with the linear relaxation of the set partitioning formulation, integrality is lost, but a very good bound is identified and it is easy to find an integer solution afterwards. The algorithm developed is limited to problems of under 200 trips because of the computer memory used. Larger problems could however be solved by using a simplex algorithm which stores the inverse basis in product form.

The network relaxation method with branching on the time intervals can solve large problems. However, it is limited to problems with relatively narrow time windows. This method is suitable for applications with small time windows where a reduction in the number of vehicles by more than 10% is not to be expected as the schedules of the trips cannot be greatly modified.

REFERENCES

BELMORE, MALONE (1971), "Pathology of Travelling Salesman Subtour Elimination Algorithm", Operations Research 19, 278-307.

BODIN, GOLDEN, ASSAD, BALL (1981), "Routing and Scheduling of Vehicules and Crews: The State of the Art", Computer and Operations Research 10, 69-211.

CARPANETO, TOTH (1980), "Some New Branching and Bounding Criteria for the Asymetric Traveling Salesman Problem", Management Science 26(7), 736-743.

CRISTOFIDES, MINGOZZI, TOTH (1981), "State-Space Relaxation Procedure for the Computation of Bounds to Routing Problems" Networks 11, 145-164.

DESROSIERS, PELLETIER, SOUMIS (1983), "Plus court chemin avec contraintes d'horaires", RAIRO, rech.opér. 17(4), 1-21.

DESROSIERS, SOUMIS, DESROCHERS (1983), "Routing with Time Windows by Column Generation", Networks, to appear.

DESROSIERS, SOUMIS, DESROCHERS, SAUVE (1983), "Routing and Scheduling with Time Windows Solved by Network Relaxation and Branch-and-Bound on Time Variables", Computer Scheduling of Public Transport, vol II, North Holland, J.-M. Rousseau ed., to appear.

FISHER, JAIKUMAR (1981), "A Generalized Assignment Heuristic for Vehicle Routing", Networks 11, 109-124.

LAND, POWELL (1973), "Fortran Codes for Mathematical Programming: Linear, Quadratic and Discrete", Wiley.

LEVIN (1971), "Scheduling and Fleet Routing Models for Transportation Systems", Transportation Science 5, 232-255.

MILLER, TUCKER, ZEMLIN (1960), "Integer Programming Formulation of Travelling Salesman Problem", Journal of the Association of Computing Machinery 7, 326 332.

SOUMIS, DESROSIERS, DESROCHERS (1983), "Optimal Urban Bus Routing with Scheduling Flexibilities", Proceedings of the 11th IFIP Conference on System Modelling and Optimization, P. Thoft-Christensen ed., to appear.

SWERSEY, BALLARD (1982), "Scheduling School Buses", Yale School of Organization and Management, presented at ORSA/TIMS Conference, San Diego, California, 1982.